2024 IEEE International Conference on Image Processing (ICIP) | 979-8-3503-4939-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICIP51287.2024.10648090

RESNERF-PCAC: SUPER RESOLVING RESIDUAL LEARNING NERF FOR HIGH EFFICIENCY POINT CLOUD ATTRIBUTES CODING

[1]Sajid Umair, [1]Birendra Kathariya, [1]Zhu Li, [2]Anique Akhtar, [2]Geert Van der Auwera

[1]University of Missouri-Kansas City, Missouri, USA
[2]Qualcomm Technologies Inc. California, USA

ABSTRACT

A point cloud (PC) is a popular 3D data representation that poses challenges due to its size, dimensionality, and unstructured nature. This paper introduces the Residual Neural Radiance Field for Point Cloud Attribute Coding (ResNeRF-PCAC), a novel approach for point cloud attribute compression. ResNeRF-PCAC combines sparse convolutions with neural radiance fields, to create a highly efficient attribute coding solution. It initially downscales the point cloud to generate a coarse thumbnail point cloud and encodes it using the G-PCC attribute encoder. The thumbnail PC is upsampled using a super-resolution network to generate a recolored PC. Color attribute residuals are then computed between the original and the super-resolved recolored PC. A ResNeRF network is employed to predict these residuals. The trained ResNeRF weights are compressed into a bitstream. The thumbnail bitstream and the compressed model weights are then transmitted to the decoder. Sparse convolution-based super-resolving network weights are shared and common across all content and need not to be signaled. Experiments on the MPEG-8i dataset demonstrate superior performance in terms of reconstruction quality and compression ratio compared to G-PCC-RAHT and G-PCC-Predlift for both v14 and v21.

Index Terms— 3D Point Cloud, Attributes Compression, Deep Learning, NeRF, Model Compression

1. INTRODUCTION

Point cloud is a collection of 3D data points that represent the shape and position of objects in a real space. Each data point has its own (x, y, z) geometry coordinates and their associated attributes such as color (r, g, b), reflectance, etc. 3D point clouds are widely used in many fields such as augmented and virtual reality AR/VR, gaming, autonomous driving, animations, and robotics. The rapid progress of 3D sensing and capturing technology has enabled applications like immersive video technology that require dense point clouds suitable for human viewing. These point clouds often contain millions of points per frame [1] that require efficient compression schemes for storage and transmission. Point cloud compression includes geometry compression as well as attribute compression. In this paper, we propose a point cloud attribute encoding scheme. Traditionally, different well-known approaches are used for point cloud attributes compression which is based on rules-based transformation such as Region-Adaptive Hierarchical Transform (RAHT) [2], hierarchical nearest neighbor prediction based Lifting Transform [3], and Graph Fourier Transform (GFT) [4]. Due to its high efficiency, RAHT is adopted by MPEG G-PCC. Zhang et al. [5] introduced a geometric coordinate-based graph and suggested compressing point cloud attributes with a graph transform but this method is not well optimized and leads to a subgraph issue. Queiroz et al. [6] developed an RAHT-based method for attribute compression and employed hierarchical subband transform to compress the point cloud attributes. Sheng et al. [7] proposed the deep learning-based end-to-end framework for point cloud attribute compression and utilized an autoencoder to encode and decode the point cloud attributes using geometry but the performance of this approach is worse compared to G-PCC with a significant loss. Fang et al. [8] proposed the learning-based attributes compression method called 3DAC that converts attributes to transform coefficients and uses a deep entropy model to predict the probabilities of these coefficients and generate an attributes bitstream but they only improve RAHT rather than achieve transform based compression of attributes. Quach et al. [9] map the point cloud attributes to a 2D grid and perform compression to generate the compressed bitstream. However, the efficiency of coding suffers and the generalization of the model is very difficult and also for individual samples, the mapping function is overfitted. Liu et al. [10] proposed a hybrid compression framework and used virtual and adaptive sampling-based sparse representation strategy, discrete cosine transform, and k-d tree decomposition for color attributes compression. Wang et al. [11] proposed a sparse tensor-based variational autoencoder (VAE) framework for point cloud attribute compression. Rodrigo et al. [12] developed a normalizing flow using the sparse convolutions-based solution for attribute compression but at high bitrates the performance of this method is worse compared to the latest G-PCC. Neural Radiance Field (NeRF) [13] has gained tremendous

This work is supported by NSF RINGs Project grant 2148382.

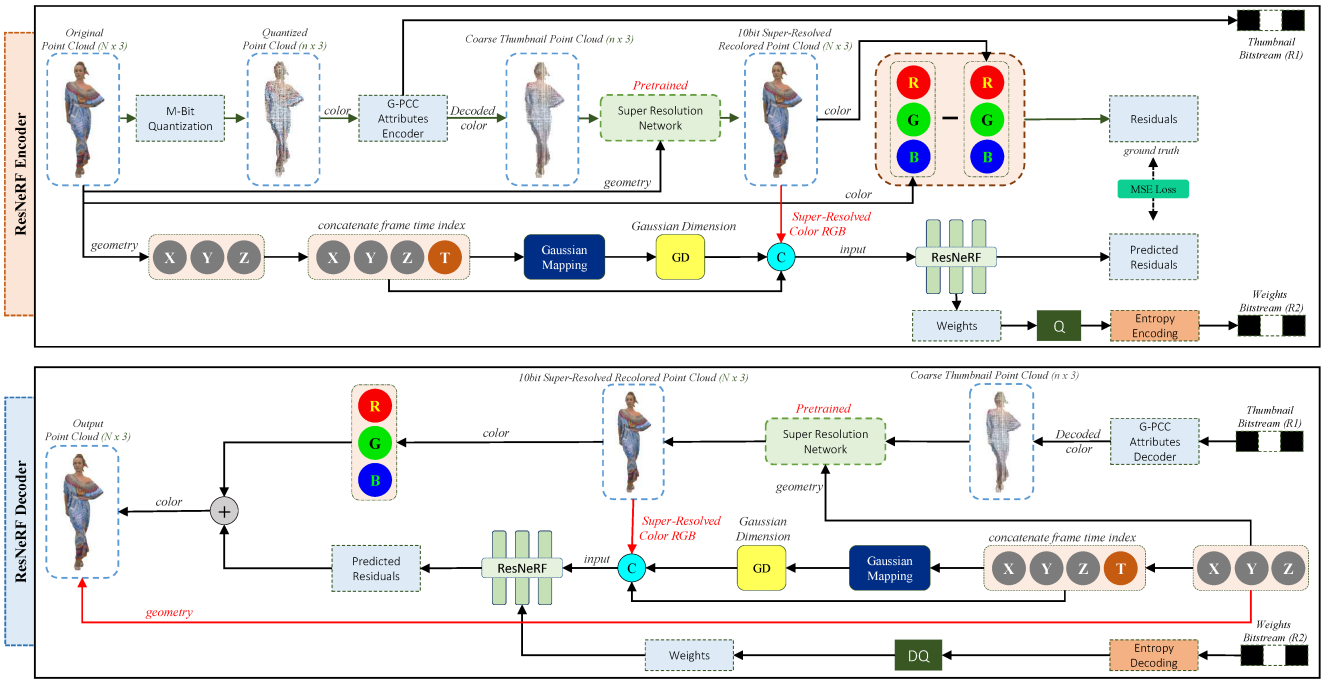

Fig. 1: Overall architecture and workflow of proposed ResNeRF-PCAC system including Encoder and Decoder for 3D point cloud attributes compression. **Q** is Quantization and **DQ** is Dequantization. **-** sign is used for subtraction in the residual calculation block. **X,Y,Z** are point cloud geometry coordinates and **T** is point cloud frame time index. **c** represents concatenation.

popularity in the field of computer vision. Mildenhall et al. [14] proposed a NeRF-based method for view synthesis and representing scenes. Kerbl et al. [15] used 3D Gaussian splatting for real-time radiance field rendering and high-quality view synthesis. Muller et al. [16] presented an instant neural graphics primitives-based solution using multiresolution hash encoding, a hash grid, and an occupancy grid to achieve high-quality neural graphics primitives. Wei et al. [17] proposed a super-resolution neural operator solution which is a deep operator learning framework used to improve the quality of low-resolution images to high-resolution images. Lu et al. [18] proposed a deep learning-based hybrid model in which they used Transformer and CNN for image super-resolution. Damodaran et al. [19] proposed an image decoder called RQAT-INR, which is based on Implicit Neural Representation(INR), in which they compressed the neural network weights. Based on all the aforementioned points and inspired by NeRF, we built our proposed novel system for point cloud attributes compression. In this paper, we propose a novel deep learning-based system for point cloud attributes compression termed *ResNeRF-PCAC*. The proposed system offers the following major contributions toward the point cloud attributes compression:

- We propose a deep learning-based system for point cloud attributes compression which will encode and decode the point cloud attributes using residual learning-based NeRF called *ResNeRF* and sparse-conv based super-resolving network.
- We apply quantization to downscale the original point cloud and then encode the downscaled point cloud using G-PCC attribute encoder to generate a coarse thumbnail point cloud. A sparse convolution based super-resolution network is employed to upscale the thumbnail PC to generate the super-resolved recolored point cloud.
- A ResNeRF network is employed to predict the RGB residuals between the original and the recolored point cloud. The trained ResNeRF weights are compressed into a bitstream.
- Finally, the ResNeRF weight bitstream and the thumbnail PC bitstream transmitted to the decoder. Sparse convolution based super-resolving network weights are shared and common across all content and need not to be signaled.

2. PROPOSED RESNERF-PCAC SYSTEM

In this section, we describe the proposed system "ResNeRF-PCAC" for point cloud attribute compression which is depicted in Fig. 1. The focus of this work is on point cloud

attribute compression and the geometry is assumed to be losslessly transmitted separately and available at the decoder.

2.1. ResNeRF-PCAC Encoder

The overall workflow of the proposed Encoder is illustrated in the top of Fig. 1.

2.1.1. Coarse Thumbnail Point Cloud

Given a point cloud with geometry coordinates $X = \{x_1, x_2, .., x_N\}$, with $x_i \in \mathbb{R}^3$ and color attributes $Y = \{y_1, y_2, ..., y_N\}$, with $y_i \in \mathbb{R}^3$. N is the total number of points in the original point cloud. We first apply M-bit quantization on geometry to downscale the original point cloud and generate the quantized point cloud with geometry coordinates $\hat{X} = \{\hat{x_1}, \hat{x_2}, ..., \hat{x_n}\}$, with $\hat{x_i} \in \mathbb{R}^3$ and color attributes $\hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_n}\}$, with $\hat{y_i} \in \mathbb{R}^3$. n is the total number of points in the quantized downscaled point cloud. Afterward, G-PCC (TMC13-v21.0) is employed to encode the attribute of the quantized point cloud using RAHT in a lossy manner with the highest rate point ($R4$). As a result, we get a compressed color bitstream for the thumbnail point cloud which is labeled **R1** in Fig. 1. The Thumbnail bitstream **R1** is transmitted to the decoder. In our experiments, we used M=1 to downscale the original 10-bit point cloud and generate a 9-bit point cloud.

2.1.2. Super-Resolving

A pretrained super-resolution network is employed to upscale the thumbnail point cloud to generate a 10-bit super-resolved recolored point cloud. The super-resolution network is shown in Fig. 2. The goal of the super-resolution network is to improve the recoloring performance of 9-bit color to 10-bit point cloud. It utilizes sparse convolution and utilizes sparse tensors $T = (C, F)$ where C is the coordinate and F is the feature. The network is designed to operate at two scales of point cloud: 10-bit and 9-bit scales. The 10-bit point cloud is recolored using the decoded color at a 9-bit scale through devoxelization. Since a 9-bit point cloud is created through voxelization from the 10-bit point cloud, devoxelization can be achieved through reverse indexing. The point cloud at 9-bit and 10-bit are first processed with their respective "head"s to expand the feature from 3 channels to 64 channels. The head block consists of **"conv→BN→ReLU→conv"**. Then output in the 9-bit branch is processed with 8 ResBlocks, whereas in the 10-bit branch, the output is processed with 2 ResBlocks. A ResBlock consists of **"conv→BN→ReLU→conv"** is shown in Fig. 2(a). The output from the ResBlock in the 9-bit branch is then upscaled to a 10-bit scale using the "un-pool" operation illustrated in Fig. 2(b). The un-pool layer, through devoxelization operation, expands the feature at a 9-bit scale to a 10-bit scale geometry. The output from the ResBlock at a 10-bit scale is then concatenated with the output from the "un-pool" layer as they both are at a 10-bit scale. The concatenated features are then fused using

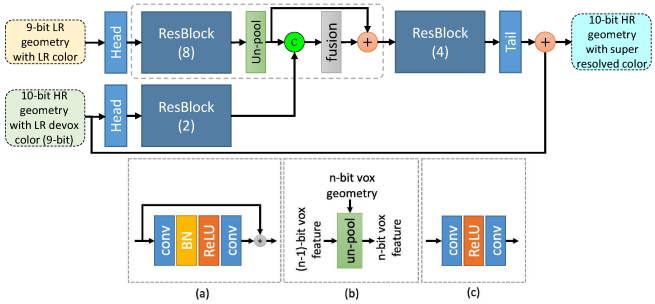

Fig. 2: Super Resolution Network. (a) ResBlock (b) Unpool Block (c) Fusion Block. Head block consists of "conv→BN→ReLU→conv" and Tail block consists "conv→ReLU→conv". BN is batch normalization, conv is the convolutional layer and ReLU is the activation function.

a fusion block which consists of **"conv→ReLU→conv"** as shown in Fig. 2(c). The output from the fusion block is added with the output from the "un-pool" layer as a residual connection. The output is again processed with 4 ResBlocks and then with a "tail". The "tail" block also consists of **"conv→BN→ReLU→conv"** however it reduces the channel size from 64 channels to 3 channels. **BN** is batch normalization. The output from the tail is finally added with the 10-bit point cloud input as a global-level residual connection and finally, the 10-bit super-resolved recolored point cloud is generated. We trained and validated the super-resolution network on THUman-2.0 and tested on the 8i Voxelized Full Bodies(8iVFBv2) dataset thats why the super-resolution network is not the part of the bitstream and we are assuming that the super-resolution network is available on the decoder side. The other reason is that the super-resolving network weights are shared and common across all content and need not to be signaled. After super-resolving, the difference in the color (r, g, b) of the original point cloud and the super-resolved recolored point cloud gives us the residual attributes.

2.1.3. Gaussian Mapping

The geometry coordinates of the original point cloud is concatenated with the time index of the frame to generate $(x, y, z, \mathbf{T})$. Gaussian Mapping [20] is applied to the concatenated geometry using the equation 1 to convert $(x, y, z, \mathbf{T})$ to a higher frequency representation called gaussian dimension.

$$\gamma(v) = [cos(2\pi Bv), sin(2\pi Bv)]^T \quad (1)$$

where $B \in \mathcal{R}^{m \times d}$ is samples from $\mathcal{N}\left(0, \sigma^2\right)$, T is transpose, $\mathcal{N}$ represents the normal distribution, σ^2 is variance and v is the original point cloud geometry with frame time index $(x, y, z, \mathbf{T})$. We set random seed with 0, m=4 and d is Gaussian dimension which we set different for different ResNeRF which is given in Table 1.

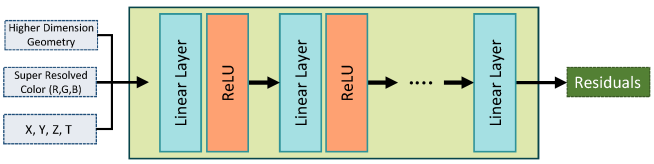

Fig. 3: Architecture of ResNeRF.

2.1.4. *ResNeRF*

Residual Neural Radiance Field (ResNeRF) is a fully connected Multi-Layer Perceptron (MLP) demonstrated in Fig. 3. ResNeRF is employed to predict the attribute residuals between the super-resolved recolored point cloud and the original point cloud. The input to ResNeRF is the concatenation of high-dimensional gaussian representation, super-resolved recolored point cloud color (RGB), and time-indexed geometry $(x, y, z, \mathbf{T})$. The output of ResNeRF is the predicted residuals. We placed the ReLU layer after each Linear layer except the last one. The Mean Squared Error (MSE) loss in equation 2 is employed for training the ResNeRF.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2}$$

where N is total number of points, y_i is target value and $\hat{y}_i$ is predicted value.

2.1.5. *ResNeRF Compression*

As explained in section 2.1.4, a ResNeRF is trained to learn the residual attributes. The ResNeRF's trained weights are extracted and an $n-bit$ scalar quantization is applied to them using equation 3. The quantized weights undergo encoding using the PAQ entropy encoder, resulting in the creation of bitstream **R2**. This bitstream is then transmitted to the decoder. On the decoder, PAQ entropy decoder is employed, and de-quantization is performed using the equation 4.

$$Z_{quant} = round\left(S * \left(\frac{Z - min}{max - min}\right) - t\right) \tag{3}$$

$$Z_{dequant} = (max - min) * \left(\frac{Z_{quant} + t}{S}\right) + min \tag{4}$$

where S is a scalar factor, Z is the value in the tensor, min, and max are the minimum and maximum values in the tensors, while t is the translation factor.

2.2. ResNeRF-PCAC Decoder

The ResNeRF-PCAC decoder is illustrated in Fig. 1. All the decoding steps are performed in an inverse order at the decoder to generate the reconstructed point cloud. As discussed in the encoding part, the thumbnail bitstream **R1** and ResNeRF MLP weights bitstream **R2** are transmitted to the decoder. These bitstreams are decoded. **R1** bitstream is decoded with G-PCC attributes decoder to generate the thumbnail point cloud color. The pre-trained super-resolution network is employed to upscale the thumbnail point cloud to generate the super-resolved recolored point cloud. The **R2** bitstream is decoded with the PAQ entropy decoder, de-quantization is applied to the decoded bitstream to generate the model weights, and then the model weights are assigned to the ResNeRF network. As original geometry is already available on the decoder side, the frame time index is concatenated to the original point cloud geometry coordinates $(x, y, z, \mathbf{T})$. Gaussian mapping is applied to convert $(x, y, z, \mathbf{T})$ to a higher frequency representation similar to that in the encoder. The higher dimensional gaussian representation, time-indexed geometry $(x, y, z, \mathbf{T})$, and the super-resolved recolored point cloud color RGB is concatenated and fed to the ResNeRF to obtain the predicted residuals. Finally, we add the predicted residuals with the super-resolved recolored point cloud color RGB to generate the final color and reconstruct the output point cloud.

3. EXPERIMENTAL SETUP AND RESULTS

3.1. Dataset

We used THUman-2.0 and 8i Voxelized Full Bodies(8iVFB v2) datasets for our experiments. From the 8i dataset, we used different point cloud sequences such as; *Soldier*, *Redandblack*, *Longdress*, and *Loot*.

3.2. Training

3.2.1. *Super Resolution Network Training*

We trained the super-resolution network using the THUman-2.0 dataset, which contains 526 full-body 10-bit point cloud models. We constructed training patches from the first 511 point clouds and created validation patches from the last 15 point clouds. Using a binary tree, each original point cloud is divided into 16 leaf nodes/patches by setting the depth size=4. We used quantization to downscale every patch from 10-bit to 9-bit and encode every patch using G-PCC attributes encoder (TMC13-v21.0-RAHT) *(lossless geometry lossy attributes configuration)* to generate the low-resolution patch. Each 9-bit encoded low-resolution patch is used as an input to the super-resolution network and the original 10-bit patch is considered as a High-resolution ground truth. In total, we used 8192 patches for training and 224 for validation. For super-resolution network testing, we used the 8iVFBv2 dataset. We used soldier, redandblack, longdress, and loot point clouds sequences for testing. We used 16-point cloud frames from every point cloud sequence. For testing, first, we apply quantization to downscale the original point cloud from 10-bit to 9-bit. Second, we encode the 9-bit point cloud using G-PCC attributes encoder (TMC13-v21.0-RAHT) *(lossless geometry lossy attributes configuration)* to generate the thumbnail PC, and the thumbnail PC is fed to the super-resolution network to get the super-resolved 10-bit recolored point cloud. For super-resolution network implementation, we used PyTorch

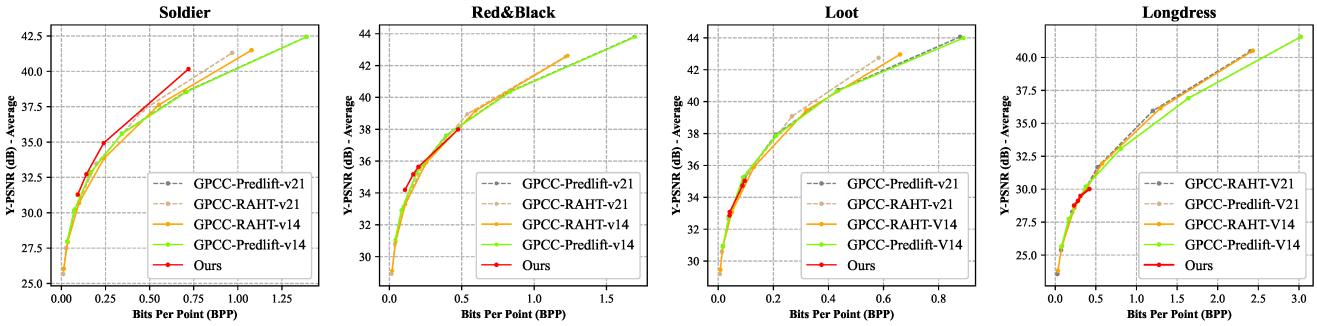

Fig. 4: Illustration of Rate-Distortion performance of proposed ResNeRF-PCAC System(Ours), G-PCC-RAHT-v21, G-PCC-Predlift-v21, G-PCC-RAHT-v14 and G-PCC-Predlift-v14.

Fig. 5: Qualitative results comparison of our proposed system with other competitors.

and Minkowski Engine [21]. Minkowski Engine is a very ideal choice for dealing with sparse tensors. We used L1 loss, Adam optimizer for network parameters optimization, and cosine-annealing learning rate scheduler. We set the learning rate between 10^{-4} to 10^{-6}. We trained the model for 30 epochs and set the batch size=2. For all convolution layers, we used Kernel size=3 and channel size=64.

3.2.2. *ResNeRF Training*

We trained the ResNeRF using the 8iVFBv2 dataset. As discussed in section 2.1.4, the input to ResNeRF is the concatenation of high-dimensional gaussian representation, super-resolved recolored point cloud color (RGB), and time-indexed geometry $(x, y, z, \mathbf{T})$ to learn and predict the residuals. We train and test ResNeRF on the first 16 frames from each point cloud sequences dataset such as soldier, redandblack, loot, and longdress. For ResNeRF testing, we used the same input like concatenation of high-dimensional gaussian representation, super-resolved recolored point cloud color (RGB), and time indexed geometry $(x, y, z, \mathbf{T})$ to the ResNeRF to get the predicted residuals. The reason to use the same input is that the same input is already available on the decoder side. Our approach is different from a traditional approach and the training and testing data are the same. For ResNeRF implementation, we used Python and PyTorch frameworks. For all ResNeRF training, we used Adam optimizer, batch size of 32000, 2000 epochs, and learning rate from 1e-3 to 1e-6.

3.3. Performance Evaluation Metrics

We used the Peak Signal-to-Noise Ratio(PSNR) to evaluate the reconstruction quality of the point cloud and used Bits Per Point(BPP) to check the compression performance. pc-error software provided by MPEG-3DGC was used to calculate the PSNR in dB of Y and YUV channels. We use Bjøntegaard Delta bit rate(BD-BR) and Bjøntegaard Delta PSNR(BD-PSNR) to measure averaged R-D performance.

3.4. Baselines

We compare the performance of our proposed ResNeRF-PCAC system with *G-PCC(RAHT)-v21*, *G-PCC(Predlift)-v21*, *G-PCC(RAHT)-v14* and *G-PCC(Predlift)-v14* provided by MPEG.

3.5. Experimental Results

In this section, we present the experimental results of our proposed system. We compare the proposed system with the baselines in terms of rate-distortion performance on the MPEG 8iVFBv2 dataset. We used different ResNeRF MLP networks with different architectures to beat different rates of the baselines with different parameters setups including Gaussian Dimension(GD), number of layers and nodes in ResNeRF, number of parameters in ResNeRF, weights quantization(Q), thumbnail predictor and model weights bitstream details and calculated Bpp and PSNR. Experimental results for Soldier, RedandBlack, Longdress, and Loot along with different parameter setups are given in Table. 1. The rate-distortion performance of our system and other methods

Table 1: ResNeRF Experimental Results.

Point Cloud Sequence	Gaussian Dimension(GD)	ResNeRF Layers	ResNeRF Nodes	No. of Parameters	Q (Bit)	Bitstream Size (Bytes) Thumbnail-R1	Bitstream Size (Bytes) Weights-R2	BPP	Y-PSNR (Avg.)
Soldier	128	3	32	5283	10	178476	4671	0.093	31.28
	128	3	128	33411	10		30671	0.142	32.72
	128	6	128	82947	10		82603	0.239	34.93
	256	6	256	329731	10		337631	0.720	40.17
RedandBlack	128	3	32	5283	10	148423	4218	0.108	34.20
	256	6	128	99331	9		84051	0.165	35.17
	512	8	128	165123	9		135301	0.201	35.64
	512	8	256	526851	10		520927	0.475	37.99
Loot	128	3	32	5283	7	62952	1311	0.0412	32.86
	128	3	32	5283	10		2230	0.0418	33.08
	256	6	128	99331	8		74703	0.088	34.73
	256	6	128	99331	9		87405	0.096	35.04
Longdress	128	6	32	8451	10	349201	8012	0.231	28.76
	512	6	64	49667	10		46258	0.280	29.12
	512	6	128	132099	10		159354	0.319	29.49
	512	8	128	165123	10		161407	0.425	30.01

are shown in Fig. 4. We can see that, our system performed better than G-PCC-RAHT-v21, G-PCC-Predlift-v21, G-PCC-RAHT-v14 and G-PCC-Predlift-v14. We visualize the reconstructed point cloud and compare it with other methods in Fig. 5. Our system improves the reconstruction quality of the point cloud with more smoothness, clarity, and the closest look toward the ground truth.

Table 2: BD-PSNR-**Y**(dB) and BD-Rate-**Y**(%) performance against other methods.

Point Cloud	Ours against GPCC-RAHT-v21		Ours against GPCC-Predlift-v21		Ours against GPCC-RAHT-v14		Ours against GPCC-Predlift-v14	
	BD-PSNR-**Y**	BD-Rate-Y	BD-PSNR-**Y**	BD-Rate-Y	BD-PSNR-Y	BD-Rate-Y	BD-PSNR-**Y**	BD-Rate-Y
Soldier	0.65	-14.65	0.87	-20.45	1.10	-23.14	0.89	-20.88
Red&Black	0.20	-5.75	0.13	-3.28	0.41	-11.24	0.15	-3.93
Longdress	-0.16	2.46	-0.10	4.08	-0.01	-0.30	-0.09	3.79
Loot	1.15	-2.49	1.06	1.39	1.38	-10.81	1.04	2.28
Average	**+0.45 dB**	**-5.10 %**	**+0.49 dB**	**-4.56 %**	**+0.72 dB**	**-11.37 %**	**+0.49 dB**	**-4.68 %**

Table 3: BD-PSNR-**YUV**(dB) and BD-Rate-**YUV**(%) performance against other methods.

Point Cloud	Ours against GPCC-RAHT-v21		Ours against GPCC-Predlift-v21		Ours against GPCC-RAHT-v14		Ours against GPCC-Predlift-v14	
	BD-PSNR-**YUV**	BD-Rate-**YUV**	BD-PSNR-**YUV**	BD-Rate-**YUV**	BD-PSNR-**YUV**	BD-Rate-**YUV**	BD-PSNR-**YUV**	BD-Rate-**YUV**
Soldier	0.42	-12.77	0.32	-9.03	1.08	-30.30	0.34	-9.78
Red&Black	-0.07	-25.01	-0.22	-20.65	0.41	-35.43	0.06	-27.83
Longdress	-0.55	23.28	-0.57	24.30	0.01	2.41	-0.55	23.75
Loot	8.62	-45.06	8.59	-50.95	9.26	-59.56	8.56	-49.97
Average	**+2.105 dB**	**-14.89 %**	**+2.03 dB**	**-14.08 %**	**+2.69 dB**	**-30.72 %**	**+2.102 dB**	**-15.95 %**

Our method BD-PSNR-Y and BD-rate-Y performance for all the sequences against the other methods are given in Table 2. We get an average 5.10% BD-rate reduction against G-PCC-RAHT-v21, 4.56% BD-rate reduction against G-PCC-Predlift-v21, 11.37% BD-rate reduction against GPCC-RAHT-v14 and 4.68% BD-rate reduction against G-PCC-predlift-v14. We gain 0.45dB improvement against GPCC-RAHT-v21, 0.49dB improvement against G-PCC-Predlift-v21, 0.72dB improvement against G-PCC-RAHT-v14, and 0.49dB improvement against G-PCC-predlift-v14. Our method BD-PSNR-YUV and BD-rate-YUV performance for all the sequences against the competitors are given in Table 3. We get an average 14.89% BD-rate reduction against G-PCC-RAHT-v21, 14.08% BD-rate reduction against G-PCC-Predlift-v21, 30.72% BD-rate reduction against G-PCC-RAHT-v14 and 15.95% BD-rate reduction against G-PCC-predlift-v14. We gain 2.105dB improvement against G-PCC-RAHT-v21, 2.03dB improvement against G-PCC-Predlift-v21, 2.69dB improvement against G-PCC-RAHT-v14, and 2.102dB improvement against G-PCC-predlift-v14.

4. ABLATION STUDY

We examine the performance of our system with different factors. A large number of layers and nodes in ResNeRF improve the PSNR and reconstruction quality of the point cloud. When we decreased the number of layers and nodes in ResNeRF, the training performance was poor. Gaussian mapping plays a very important role in this research. Using Gaussian mapping provides a significant gain in performance. We also tried to directly learn the attributes RGB color using ResNeRF but the performance of residuals learning using ReNeRF was outstanding and better that is why we consider the residual learning approach/results.

5. CONCLUSION

We proposed a point cloud attributes compression system called ResNeRF-PCAC. We downscale the original point cloud and generate a thumbnail PC. The thumbnail PC is upsampled using a super-resolution network to generate a recolored PC. We calculate RGB attribute residuals between the original and super-resolved recolored point clouds and learn the residuals using ResNeRF. We transmit the thumbnail PC and model weights bitstream to the decoder. Experimental results show that our proposed system performed better than the competitor's.

6. REFERENCES

[1] Li Li, Zhu Li, Vladyslav Zakharchenko, Jianle Chen, and Houqiang Li, "Advanced 3d motion prediction for video-based dynamic point cloud compression," *IEEE Transactions on Image Processing*, vol. 29, pp. 289–302, 2019. 1

[2] Reetu Hooda and W David Pan, "Early termination of dyadic region-adaptive hierarchical transform for efficient attribute compression of 3d point clouds," *IEEE Signal Processing Letters*, vol. 29, pp. 214–218, 2021. 1

[3] Danillo Graziosi, Ohji Nakagami, Shinroku Kuma, Alexandre Zaghetto, Teruhiko Suzuki, and Ali Tabatabai, "An overview of ongoing point cloud compression standardization activities: Video-based (v-pcc) and geometry-based (g-pcc)," *APSIPA Transactions on Signal and Information Processing*, vol. 9, pp. e13, 2020. 1

[4] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011. 1

[5] Cha Zhang, Dinei Florencio, and Charles Loop, "Point cloud attribute compression with graph transform," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 2066–2070. 1

[6] Ricardo L De Queiroz and Philip A Chou, "Compression of 3d point clouds using a region-adaptive hierarchical transform," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3947–3956, 2016. 1

[7] Xihua Sheng, Li Li, Dong Liu, Zhiwei Xiong, Zhu Li, and Feng Wu, "Deep-pcac: An end-to-end deep lossy compression framework for point cloud attributes," *IEEE Transactions on Multimedia*, vol. 24, pp. 2617–2632, 2021. 1

[8] Guangchi Fang, Qingyong Hu, Hanyun Wang, Yiling Xu, and Yulan Guo, "3dac: Learning attribute compression for point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14819–14828. 1

[9] Maurice Quach, Giuseppe Valenzise, and Frederic Dufaux, "Folding-based compression of point cloud attributes," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 3309–3313. 1

[10] Hao Liu, Hui Yuan, Qi Liu, Junhui Hou, Huanqiang Zeng, and Sam Kwong, "A hybrid compression framework for color attributes of static 3d point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 3, pp. 1564–1577, 2022. 1

[11] Jianqiang Wang and Zhan Ma, "Sparse tensor-based point cloud attribute compression," in *2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2022, pp. 59–64. 1

[12] Rodrigo B Pinheiro, Jean-Eudes Marvie, Giuseppe Valenzise, and Frédéric Dufaux, "Nf-pcac: Normalizing flow based point cloud attribute compression," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5. 1

[13] Kyle Gao, Yina Gao, Hongjie He, Denning Lu, Linlin Xu, and Jonathan Li, "Nerf: Neural radiance field in 3d vision, a comprehensive review," *arXiv preprint arXiv:2210.00379*, 2022. 1

[14] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021. 2

[15] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, 2023. 2

[16] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022. 2

[17] Min Wei and Xuesong Zhang, "Super-resolution neural operator," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18247–18256. 2

[18] Zhisheng Lu, Juncheng Li, Hong Liu, Chaoyan Huang, Linlin Zhang, and Tieyong Zeng, "Transformer for single image super-resolution," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 457–466. 2

[19] Bharath Bhushan Damodaran, Muhammet Balcilar, Franck Galpin, and Pierre Hellier, "Rqat-inr: Improved implicit neural image compression," *arXiv preprint arXiv:2303.03028*, 2023. 2

[20] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7537–7547, 2020. 3

[21] Christopher Choy, JunYoung Gwak, and Silvio Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3075–3084. 5